```
Start coding or generate with AI.
```

```
#load the csv files data

import pandas as pd
import matplotlib.pyplot as plt

customers=pd.read_csv("Customers.csv")
products=pd.read_csv("Products.csv")
transactions = pd.read_csv("Transactions.csv")


#print the data

print(transactions.head())
print(transactions.info())
print(transactions.describe())

transactions ['TransactionDate'] = pd.to_datetime(transactions['TransactionDate'], errors='coerce')

invalid_dates = transactions[pd.to_datetime(transactions['TransactionDate'], errors='coerce').isna()]
print(invalid_dates)

print(transactions['TransactionDate'].dtype)

#checking for duplicates
#for customers
print(customers.duplicated().sum())

print(customers['SignupDate'].min(),customers['SignupDate'].max())


#for products
print(products[products['Price'] < 0])


#for transactions
invalid_transactions = transactions[transactions['TotalValue']!=transactions['Quantity']* transactions['Price']]
print(invalid_transactions)



# Convert TransactionDate to datetime if not already done transactions['TransactionDate'] = pd.to_datetime(transactions['TransactionDate'
# Group transactions by month/year
transactions ['YearMonth'] = transactions['TransactionDate'].dt.to_period('M')
transaction_counts = transactions['YearMonth'].value_counts().sort_index()

# Plot the trend
plt.figure(figsize=(10,6))
transaction_counts.plot(kind='line' )
plt.title('Monthly Transactions Over Time')
plt.xlabel('Year-Month')
plt.ylabel('Number of Transactions')
plt.show()


# Analyze region distribution
region_counts = customers['Region'].value_counts()
print(region_counts)
region_counts.plot(kind='bar', title='Customer Distribution by Region')
plt.show()

# Analyze signup trends
customers['SignupDate'] = pd.to_datetime(customers['SignupDate'])
signup_trends = customers['SignupDate'].dt.to_period('M').value_counts().sort_index()
signup_trends.plot(kind='line', title='Customer Signup Trends')
plt.show()

# Merging datasets
merged_data = transactions.merge(customers, on='CustomerID', how='left').merge(products, on='ProductID', how='left')

# Inspect the merged data
print(merged_data.info())
print(merged_data.head())

# Calculate total revenue by product category
category_revenue = merged_data.groupby('Category')['TotalValue'].sum().sort_values(ascending=False)
print(category_revenue)

# Plot the revenue contribution by category
category_revenue.plot(kind='bar', figsize=(10, 6), title='Revenue by Product Category')
```

```
category_revenue.plot(kind='bar', figsize=(10, 6), title='Revenue by Product Category')
plt.ylabel('Total Revenue (USD)')
plt.show()

# Calculate the most sold product in each category
top_products = merged_data.groupby(['Category', 'ProductName'])['Quantity'].sum().reset_index()
top_products = top_products.sort_values(['Category', 'Quantity'], ascending=[True, False]).groupby('Category').head(1)
print(top_products)


# Revenue by region
region_revenue = merged_data.groupby('Region')['TotalValue'].sum().sort_values(ascending=False)
print(region_revenue)

# Plot region-wise revenue distribution
region_revenue.plot(kind='pie', autopct='%1.1f%%', figsize=(8, 8), title='Revenue by Region')
plt.ylabel('')
plt.show()

# Calculate customer lifetime value
customer_ltv = merged_data.groupby('CustomerID')['TotalValue'].sum().sort_values(ascending=False)
print(customer_ltv.head(10))

# Plot top customers by total revenue
customer_ltv.head(10).plot(kind='bar', figsize=(10, 6), title='Top 10 Customers by Total Revenue')
plt.ylabel('Total Revenue (USD)')
plt.xlabel('CustomerID')
plt.show()


# Monthly revenue trends
merged_data['TransactionMonth'] = merged_data['TransactionDate'].dt.to_period('M')
monthly_revenue = merged_data.groupby('TransactionMonth')['TotalValue'].sum()

# Plot monthly revenue trends
monthly_revenue.plot(kind='line', figsize=(10, 6), title='Monthly Revenue Trend')
plt.ylabel('Revenue (USD)')
plt.xlabel('Month')
plt.show()
```

```
0        T00001      C0199      P067  2024-08-25 12:38:23         1
1        T00112      C0146      P067  2024-05-27 22:23:54         1
2        T00166      C0127      P067  2024-04-25 07:38:55         1
3        T00272      C0087      P067  2024-03-26 22:55:37         2
4        T00363      C0070      P067  2024-03-21 15:10:10         3

   TotalValue   Price
0      300.68  300.68
1      300.68  300.68
2      300.68  300.68
3      601.36  300.68
4      902.04  300.68
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 7 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   TransactionID    1000 non-null   object
 1   CustomerID       1000 non-null   object
 2   ProductID        1000 non-null   object
 3   TransactionDate  1000 non-null   object
 4   Quantity         1000 non-null   int64
 5   TotalValue       1000 non-null   float64
 6   Price            1000 non-null   float64
dtypes: float64(2), int64(1), object(4)
memory usage: 54.8+ KB
None
          Quantity   TotalValue       Price
count  1000.000000  1000.000000  1000.00000
mean      2.537000   689.995560   272.55407
std       1.117981   493.144478   140.73639
min       1.000000    16.080000    16.08000
25%       2.000000   295.295000   147.95000
50%       3.000000   588.880000   299.93000
75%       4.000000  1011.660000   404.40000
max       4.000000  1991.040000   497.76000
Empty DataFrame
Columns: [TransactionID, CustomerID, ProductID, TransactionDate, Quantity, TotalValue, Price]
Index: []
datetime64[ns]
0
2022-01-22 2024-12-28
Empty DataFrame
Columns: [ProductID, ProductName, Category, Price]
Index: []
    TransactionID CustomerID ProductID     TransactionDate  Quantity  \
17         T00270      C0101      P034 2024-11-07 02:48:08         3
29         T00218      C0148      P057 2024-01-17 19:40:55         3
30         T00417      C0035      P057 2024-04-20 22:54:54         3
31         T00492      C0120      P057 2024-08-08 05:40:02         3
35         T00703      C0092      P057 2024-02-04 00:31:54         3
..            ...        ...       ...                 ...       ...
924        T00823      C0095      P079 2024-09-30 10:45:06         3
935        T00660      C0057      P008 2024-09-23 16:46:01         3
946        T00646      C0036      P091 2024-01-23 12:53:51         3
947        T00793      C0054      P091 2024-10-07 17:48:28         3
948        T00798      C0015      P091 2024-09-21 01:39:03         3

     TotalValue   Price
17       651.15  217.05
29       719.10  239.70
30       719.10  239.70
31       719.10  239.70
35       719.10  239.70
..          ...     ...
924     1252.11  417.37
935      440.55  146.85
946      668.85  222.95
947      668.85  222.95
948      668.85  222.95

[88 rows x 7 columns]
```